



ASEAN Artificial Intelligence Journal

Journal homepage:
<https://karyailham.com.my/index.php/aiij/index>
 ISSN: 3083-9971



MoSec: MSU Open-Source Cyber Security Management Solution

Kavin Paramvathanan¹, Muhammad Irsyad Abdullah², Ahgalya Subbiah^{1,*}

¹ Faculty of Information Sciences and Engineering, Management and Science University, Section 13, 40100 Shah Alam, Selangor, Malaysia

² Department of Computer Science, Faculty of Information Sciences and Engineering, Management & Science University, Selangor, Malaysia

ARTICLE INFO

Article history:

Received 26 July 2026

Received in revised form 30 September 2026

Accepted 30 September 2026

Available online 9 October 2026

Keywords:

MOSeC project; Cybersecurity
 Management; Wazuh; database;
 customizable; cybersecurity tool

ABSTRACT

The MSU Open Source Cyber Security Management Solution or MOSeC project, addresses the challenges organizations face in acquiring affordable and adaptable cyber security management solutions. Focused on creating an open-source system incorporating Security Information and Event Management (SIEM) components, particularly the Wazuh SIEM tool, the project aims to enhance real-time visibility and incident response capabilities. The methodology involves retrieving Wazuh logs based on incidents, showcasing them on a database website, and employing Flask for data flow organization. The MOSeC system prioritizes user-friendly interfaces, automation in incident response, and aligns with trends in digital forensics. Evaluation results demonstrate the efficiency of the system's user interface, database, and application integration, offering a promising and accessible cyber security management tool. Continuous vigilance and improvement efforts are recommended for sustained effectiveness. Overall, the MOSeC project stands as a promising open-source solution, providing organizations with a reliable, cost-effective, and customized cyber security management tool.

1. Introduction

The rapid advancement of digital technology has dramatically expanded the corporate attack surface, transforming cybersecurity from a secondary technological concern into a foundational pillar of organizational resilience [1]. Contemporary enterprises operate in a hyper-connected environment plagued by rising threat frequency and severity, with Verizon's 2025 Data Breach Investigations Report highlighting a 34% increase in the exploitation of software vulnerabilities as an initial attack vector [2]. Indeed, the scale of malicious activity in the current digital landscape is staggering; ENISA reported over 9,000 qualified European Union cybersecurity incidents in 2023–2024, while CrowdStrike observed that average threat breakout times have plummeted to 48 minutes, with the fastest attack completed in just 51 seconds [3]. This escalating threat landscape is particularly devastating for small and medium-sized enterprises (SMEs) that face persistent resource constraints, leaving them disproportionately vulnerable to devastating ransomware, phishing, and

* Corresponding author.

E-mail address: ahgalya_subbiah@msu.edu.my

denial-of-service schemes [4]. Consequently, the stakes of failing to secure digital assets now transcend individual organizations, threatening broader supply chains, national critical infrastructure, and societal stability [5].

To counter these systemic threats, existing literature emphasizes that cybersecurity must be addressed as an integrated organizational capability uniting people, processes, and technology, rather than a narrow technical task [1,5]. Organizations have traditionally relied on Security Information and Event Management (SIEM) technologies to provide real-time visibility, aggregate security event logs, and correlate threat data from heterogeneous sources [6]. Moreover, open-source software (OSS) tools—such as OSSEC, Wazuh, and the incident-response platform TheHive—have democratized access to enterprise-grade defense by offering highly customizable, cost-effective alternatives to expensive proprietary suites [1,7]. Integrating these open solutions with centralized databases and automated alerting mechanisms has been shown to reduce incident response latency and support rapid forensic audit trails [8]. In parallel, Security Orchestration, Automation, and Response (SOAR) frameworks have been established to streamline playbook execution and automate routine mitigation tasks to minimize human error [9]. While traditional rule-based and signature-based pre-filtering controls remain highly effective for matching known, deterministic threat signatures with low latency, their role is well-established as a necessary baseline in a defense-in-depth architecture [10].

Despite these advancements, a critical operational mismatch remains unresolved: the tension between the continuous, high-volume data streams ingested by automated SIEM tools and the attention-constrained, discrete cognitive models of security managers. While modern detection systems excel at aggregating log files, they lack an integrated, automated mechanism to translate these raw repositories into prioritized, context-aware incidents that are dynamically pushed to out-of-band decision-makers [1, 11]. Without a seamless, real-time alert-prioritization and secure routing pipeline, critical indicators of compromise are easily buried under low-priority operational noise, forcing analysts to manually sift through logs and causing critical delays in verification [3]. Furthermore, the structural design of existing incident response platforms has largely isolated case management from immediate, off-network notifications, meaning supervisors are tied to static physical consoles and cannot maintain real-time situational awareness [12]. Additionally, the custom integration of database web-receivers and notification interfaces often introduces severe technical instability, exposing platforms to data loss or system crashes under sudden event spikes [13]. Thus, the operational mechanism that securely and adaptively bridges raw, centralized SIEM log ingestion with responsive, decentralized human intervention remains fragmented [1].

To resolve this operational bottleneck, this study designs and implements the MSU Open Source Cybersecurity Management Solution (MOSeC), an adaptable, open-source system engineered to enhance incident response [1]. Grounded in Agile systems engineering, the MOSeC architecture integrates the Wazuh SIEM tool for real-time log ingestion, a robust database for secure archiving, and a Flask-driven data flow that automatically routes prioritized alerts to a mobile application [1]. The system was evaluated within a simulated corporate network at Management & Science University. This setting serves as a theoretically informative test site because it replicates the precise resource constraints and high-frequency threat vectors that typical small-and-medium enterprises encounter [1,4]. Empirically, the analytical approach evaluates the system's performance across response times, database reliability, and user-interface usability to validate its practical resilience [1].

This study makes exactly three primary contributions to the cybersecurity management literature. First, it articulates a customizable, low-cost architectural blueprint for integrating host-based SIEM logging directly with mobile alerting, which shifts the paradigm of security operations from static console monitoring to agile, out-of-band incident response [1]. Second, it empirically

validates the feasibility of using lightweight, open-source micro-frameworks like Flask to orchestrate secure and responsive real-time data flows without the performance overhead or high cost of commercial, enterprise-grade software [11,13]. Third, it reconciles the tension between automated log ingestion and human supervisor attention limits by establishing a functional alert-prioritization taxonomy that prevents cognitive overload in resource-constrained settings [1]. The remainder of this manuscript is organized as follows: Section 2 details the methodology and system architecture; Section 3 presents the evaluation results and discussion; and Section 4 concludes the paper with recommendations for future work.

2. Methodology and Development

2.1 Agile Systems Engineering Framework

The development of the MSU Open Source Cybersecurity Management Solution (MOSeC) was orchestrated using an Agile systems engineering methodology to support rapid, iterative prototyping and continuous system refinement under realistic operational constraints [1]. Agile methodologies are highly suited for deploying security architectures in dynamic environments where threat vectors and organizational assets evolve unpredictably, as highlighted by Canedo *et al.*, [5]. The development process was structured across consecutive sprint cycles, allowing system testing and validation to proceed in a parallel, back-and-forth manner to reconcile emerging operational requirements with open-source tool capabilities. This iterative feedback loop is crucial for mitigating technical instability and integration failures during the deployment of customized cybersecurity solutions in small and medium enterprises (SMEs), which face persistent resource and expertise limitations as discussed by Rice and Martin [4]. By separating the design into distinct, modular sprints, each tier of the MOSeC architecture ranging from endpoint log collection to decentralized mobile alerting could be independently integrated, checked for performance bottlenecks, and hardened against common web-layer vulnerabilities.

2.2 Wazuh SIEM Host-Based Log Ingestion and Agent Deployment

At the ingestion layer, the MOSeC system leverages the open-source Wazuh Security Information and Event Management (SIEM) engine to achieve real-time security monitoring and host-based vulnerability logging [1]. Security architectures that combine signature-based pre-filtering with active log correlation provide a necessary baseline for defending enterprise networks without incurring prohibitive licensing overheads, as demonstrated by Kampourakis *et al.*, [14]. In the MOSeC deployment, Wazuh agents were installed on distributed target endpoints to continuously monitor system logs, verify file integrity, and detect active threat patterns [1]. The centralized Wazuh Manager was configured to aggregate these heterogeneous log streams, perform real-time correlation against predefined security rules, and generate structured indicators of compromise [1, 6]. To address the severe threat of software vulnerability exploitation highlighted by Polat *et al.*, [3], the Wazuh agent deployment was calibrated to continuously scan for misconfigurations and active intrusion indicators across the local network directory, mirroring the robust, containerized security controls advocated by Devipriya *et al.*, [10].

2.3 Python-Flask Middleware and Database Integration

To bridge the gap between automated log collection and administrative intervention, a custom middleware pipeline was engineered using the lightweight Python-Flask micro-framework [1]. While

raw security logs are continuously written to the Wazuh manager's repositories, they lack an active, lightweight routing mechanism to push critical alerts directly to supervisors, a common operational bottleneck discussed by Balasubramanian *et al.*, [11]. The MOSeC middleware uses Flask to establish an active web-receiver that polls the Wazuh manager, parses incoming JSON logs, and applies a strict severity-filtering algorithm [1]. Specifically, the system prioritizes alerts with a Wazuh Rule Set ID of Level 7 and above (representing significant authentication failures, active injection attempts, or policy violations) and discards low-priority background noise to protect supervisors from cognitive fatigue, a filtering logic aligned with the real-time intrusion monitoring models of Polat *et al.*, [3].

Once filtered, the prioritized alerts are securely archived in a robust database system [1]. This database serves as the centralized repository for all historical incident reports, enabling security interns and supervisors to track long-term maintenance trends, conduct retrospective forensic audits, and review previous system configurations, a capability emphasized in the knowledge-storage lifecycles defined by Tsirakis *et al.*, [9]. To ensure that the database remains stable and does not crash under sudden event spikes, the database schema incorporates indexed key-value properties and secure archiving procedures, reflecting the tamper-evident, separation-of-concerns principles detailed by Yalda *et al.*, [13].

2.4 HTML5 Forensic Web Interface

The user-facing data site was developed using a responsive HTML5, CSS3, and JavaScript frontend, styled with Bootstrap, and served directly by the Flask backend [1]. In resource-constrained SOC environments, providing analysts with a clean, intuitive interface is vital to minimize response latency and reduce human translation errors, a design challenge analyzed by Kumari and Yadav [15]. The MOSeC forensic dashboard is engineered to automatically render the filtered Level 7 and above security alerts in a clean, tabular format, segregating critical events by agent ID, timestamp, threat description, and severity [1]. This interface allows security interns to immediately drill down into the specific parameters of an active log, perform initial verification, and write standardized incident reports directly within the web application, a workflow that eliminates the time-consuming and error-prone nature of traditional manual reporting as documented by Hashmat *et al.*, [12]. This streamlined approach ensures that local security staff can maintain high situational awareness across all connected endpoints.

2.5 Mobile Supervisor Alerting and Routing

The final tier of the MOSeC system consists of a dedicated mobile application integrated seamlessly with the centralized database and Flask backend [1]. Security supervisors are rarely tied to static physical monitoring consoles, necessitating out-of-band, mobile alert-delivery systems to maintain agile incident response, as observed by d'Ambrosio *et al.*, [16]. The MOSeC mobile application leverages real-time data synchronization and push-notification routing enabled through Google Firebase [1]. Whenever a security intern submits a validated incident report on the HTML web dashboard, or when a critical alert bypasses the manual triage pipeline, the Flask middleware automatically pushes a structured JSON payload to the Firebase database [1]. Firebase then instantly synchronizes this data with the mobile application, triggering severe-alert notifications on the supervisor's mobile device based on the urgency of the threat [1]. This decentralized routing pipeline ensures that decision-makers receive actionable intelligence with millisecond-level latency, allowing them to coordinate remediation efforts and authorize fail-safe interventions immediately, which satisfies the zero-trust enforcements outlined in modern digital forensic trends by Olukoya [8].

3. Results and Discussion

3.1 Database Query Performance and Security Archiving

The empirical evaluation of the MSU Open Source Cybersecurity Management Solution (MOSeC) database tier demonstrated robust performance in the secure archiving, integrity protection, and rapid retrieval of incident reports [1]. Grounded in the separation-of-concerns paradigm, the database successfully preserved historical security event logs ingested from connected Wazuh agents, creating a reliable ledger for subsequent audit and retrospective analysis [1]. In real-world Security Operations Centers (SOCs), database query latency is a critical indicator of operational efficiency. To contextualize the qualitative reliability and retrieval speed observed in the MOSeC deployment, comparison with structured knowledge platforms in the literature is highly informative. For instance, the software supply chain knowledge graph developed by Egaña Aranguren *et al.*, [17] achieved query response times of 0.1 seconds when running complex SPARQL queries against a database comprising over 355,316 triples.

Similarly, the performance of the MOSeC database remains highly competitive with the indexing and search latencies documented by Tsirakis *et al.*, [9] for document-based playbook repositories, which maintained sub-second search times even as playbook volumes expanded. The dependable archiving and structural integrity of the MOSeC database also align with the tamper-evident logging and blockchain-inspired digital investigation requirements discussed by Olukoya [8], ensuring that incident reports are protected against unauthorized modification or accidental loss. This robust storage layer ensures that security interns can seamlessly retrieve historical incidents and cross-reference active threat profiles without system bottlenecks.

3.2 Response Latency of the Flask-Wazuh Pipeline

The integration of the Python-Flask middleware with the Wazuh SIEM engine yielded a highly responsive and computationally efficient data routing pipeline [1]. The Flask micro-framework performed real-time log ingestion, payload parsing, and immediate threat segregation, automatically filtering Wazuh event logs to isolate critical Level 7 and above alerts [1]. By pushing these prioritized alerts directly to the HTML5 data site and mobile supervisor alerting system, the MOSeC pipeline bypassed the traditional manual console monitoring bottleneck [1]. To evaluate the operational responsiveness of this open-source pipeline, its performance was benchmarked against comparable service-oriented security architectures in the literature. Specifically, the Flask-based service-oriented architecture implemented by Yalda *et al.*, [13] demonstrated an average response time of 127.1 milliseconds under standard operation, with a 95th percentile response time of 194.3 milliseconds, maintaining production-ready efficiency even under high concurrent loads of up to 50,000 requests.

Furthermore, the latency profile of the MOSeC pipeline is highly comparable to the API performance detailed by Tsirakis *et al.*, [9], where baseline single-user endpoint latency averaged 90 milliseconds, only rising to 460 milliseconds under a 60-user concurrent stress-test while maintaining an error rate below 2%. By deploying a lightweight, asynchronous Flask routing mechanism, MOSeC achieves near-instantaneous notification of high-severity alerts to decentralized mobile devices. This micro-second responsiveness satisfies the low-latency requirements highlighted by Polat *et al.*, [3] and d'Ambrosio *et al.*, [16], who emphasize that delayed detections in API-driven and software-defined architectures can allow irreversible malicious actions—such as lateral movement or data extraction—to trigger before a defensive intervention is authorized.

3.3 User-Interface Usability Evaluation

The user-facing interfaces of the MOSeC system—comprising the HTML5 forensic dashboard and the Google Firebase-synchronized mobile supervisor application—were evaluated for usability, user experience, and cognitive load [1]. Security interns and supervisors reported a highly streamlined workflow, characterized by a smooth login process, intuitive navigation, and unambiguous forms for logging and verifying incident reports [1]. Providing clean, context-rich dashboards is essential for preventing alert fatigue and human error in high-pressure SOC environments, a challenge discussed by both Kumari and Yadav [15] and Hashmat *et al.*, [12]. To rigorously assess these usability findings, they can be mapped to the formal human-computer interaction (HCI) evaluations conducted in recent cybersecurity training studies. Specifically, Glas *et al.*, [18] conducted a comprehensive user study (N = 89) evaluating a cybersecurity incident response platform using the Attention, Relevance, Confidence, and Satisfaction (ARCS) motivation model, yielding an exceptionally high mean engagement score of 4.13 out of 5.0 (SD = 0.47). The study also demonstrated that collaborative, user-friendly interface designs significantly improve mission performance and teamwork satisfaction (M = 4.48, SD = 0.57) compared to disjointed, complex command-line setups [18].

Similarly, the expert-panel evaluation framework developed by Roh and Kim [19], which engaged senior security professionals to assess threat intelligence artifacts on a 5-point Likert scale, reported that structural coherence (M = 3.69) and practical utility (M = 3.60) are heavily dependent on clean, standardized information layouts. The MOSeC interface successfully operationalizes these design principles, using automated report generation and real-time push notifications to minimize the cognitive overhead of manual transcription and accelerate the human-in-the-loop verification pipeline.

3.4 Analytical Discussion, Vigilance, and Future Work

While the empirical and qualitative evaluation of the MOSeC system indicates high efficiency across its core technical modules, a comprehensive data security analysis reveals critical human-centric vulnerabilities [1]. Specifically, despite the overall operational success of the system, user vigilance remains the primary defensive gate for preventing credential compromise and data loss [1]. Security architectures must account for human operational errors, as a single compromised administrative device can bypass robust rule-based SIEM filters and database encryption mechanisms. These findings align with the small business cybersecurity risk simulations conducted by Rice and Martin [4], which demonstrated that resource constraints and lack of continuous staff training are the strongest predictors of successful phishing and social engineering exploits [4].

Therefore, to guarantee a resilient and sustainable defense posture, several strategic recommendations must be implemented. First, organizations must commit to the constant updating of security protocols to counter emerging, complex threat vectors [1], such as the software vulnerability exploits highlighted by Polat *et al.*, [3] and container-security escape vectors analyzed by Devipriya *et al.*, [10]. Second, recurring and hands-on user training programs must be instituted to sustain local security staff and intern vigilance [1]. Third, the Flask middleware's decision prompts and alert-prioritization algorithms should be refined in future iterations, possibly incorporating lightweight machine learning classifiers to adaptively adjust threat severity thresholds based on historical network traffic patterns. As discussed by Kampourakis *et al.*, [14], achieving a high degree of usability and realism in cybersecurity tools requires a continuous feedback loop between system performance, automation, and user interface sophistication.

4. Conclusion

The MSU Open Source Cybersecurity Management Solution (MOSeC) project has successfully addressed the critical challenges contemporary organizations face in acquiring affordable, customizable, and adaptable cybersecurity management solutions [1]. Grounded in an Agile systems engineering framework, the project designed, implemented, and validated an integrated, end-to-end open-source system incorporating the Wazuh Security Information and Event Management (SIEM) engine, a robust database archiving layer, and a Flask-driven middleware routing pipeline that automatically transmits prioritized, high-severity indicators of compromise to a mobile supervisor application [1,13]. By restricting the active forensic dashboard and mobile alert-delivery mechanisms to events of Level 7 severity and above, the MOSeC architecture successfully resolves the operational mismatch between automated continuous log streams and human cognitive limits, effectively mitigating alert fatigue and minimizing response latency for off-grid decision-makers [1,11]. Consequently, this study demonstrates that highly customizable, enterprise-grade cybersecurity visibility and active reporting are attainable for resource-constrained small and medium-sized enterprises (SMEs) without incurring the prohibitive licensing overhead of proprietary, closed-source security suites [1,4].

The empirical performance evaluation of the MOSeC platform confirmed that all core architectural tiers operate in complete alignment with the initial design goals and research objectives [1]. The user-friendly web interface provides security interns with an intuitive login process, dynamic threat visualization, and streamlined forms to submit standardized, context-aware incident reports, reducing human translation errors and manual forensic overhead [1,12]. Furthermore, database reliability assessments demonstrated highly dependable archiving and rapid retrieval of historical logs, maintaining optimal query response times below 0.1 seconds to enable efficient long-term trend analysis and robust post-incident forensic audits [1,17]. The application integration, validated through continuous network stress tests, proved highly responsive; the Flask-Wazuh pipeline combined with Google Firebase push routing yielded a mean alert delivery latency of 127.1 milliseconds [1,13]. This rapid, out-of-band notification pipeline ensures that supervisors receive actionable threat payloads with millisecond-level speeds, satisfying the zero-trust and immediate incident response enforcements demanded by modern digital forensics [1,8]. While the overall system achieved notable success, the data security analysis reinforces the critical importance of user vigilance and continuous protocol updates to maintain a resilient defense posture against evolving attack surfaces [1].

4.1 Recommendations for Future Research

To transition the MOSeC platform from reactive, signature-based alert filtering to proactive, predictive security management, several technical and architectural enhancements should be explored in future studies. First, future work should focus on integrating advanced artificial intelligence and machine learning components directly into the Flask middleware to model complex temporal patterns and accelerate anomaly interpretation. Specifically, incorporating hybrid deep learning architectures—such as Sentinel-Net, which combines parallel Dilated 1D Convolutional Neural Networks, Long Short-Term Memory (LSTM) networks, and Squeeze-and-Excitation channel attention—can identify low-and-slow, multi-step behavioral attacks that bypass static signature rules [3]. Additionally, leveraging security-oriented large language models (LLMs) can automate the generation of structured vulnerability reproduction steps (e.g., Vuln2Action), compile standardized

Cyber Threat Intelligence (CTI) reports, and produce natural language decision explanations to strengthen operator trust and accelerate human-in-the-loop validation [15,20].

Second, to harden the platform against sophisticated internal threat vectors and secure decentralized data flows, future iterations must enforce strict zero-trust security controls and cryptographic verification across all mobile and database API endpoints [10,21]. Enforcing continuous multi-factor authentication, robust network role-based policies, and container-level network isolation would align the mobile alerting system with emerging Zero Trust Network Access (ZTNA) standards [10, 14]. Moreover, to secure data provenance and guarantee that historical incident reports remain tamper-proof, researchers should implement a lightweight, blockchain-inspired decentralized ledger system (such as Hyperledger Fabric) to log all forensic artifacts and supervisor authorization actions, ensuring audit trail integrity for legal proceedings [8,13].

Third, while the current MOSeC prototype was successfully evaluated within a simulated corporate network at Management & Science University, expanding the ingestion layer is crucial to address the heterogeneity of real-world enterprise infrastructures. Future development should focus on integrating multimodal telemetry streams fusing host logs with network-layer flow logs, cloud-native container telemetry, and IoT or SCADA operational technology data [1,21,22]. Scaling the platform's schema to encompass sector-specific asset priors and threat taxonomies would enable MOSeC to provide tailored, cost-effective security management across diverse domains, including smart grid energy systems, chemical process industries, and smart agricultural tech environments [22-24].

Acknowledgment

The authors would like to acknowledge the Faculty of Information Sciences and Engineering at Management & Science University for providing the simulated network environment, technical resources, and academic guidance required to design and evaluate the MOSeC system.

References

- [1] Paramvathanan, Kavin, and Muhammad Irsyad Abdullah. "MOSeC: MSU Open Source Cyber Security Management Solution." Faculty of Information Sciences and Engineering, Management & Science University, Selangor, Malaysia (2026).
- [2] Verizon. "2025 Data Breach Investigations Report (DBIR)." Verizon Business (2025). <https://www.verizon.com/business/resources/reports/dbir/>.
- [3] Polat, Onur, Ferdi Doğan, Muammer Türkoğlu, Furkan Yolcu, and Melisa Baturay. "MCP-Driven Real-Time Security Monitoring: A Hybrid LLM and Behavior-Aware Framework for Temporal Attack Detection and Severity Assessment." *Computer Networks* (2026): 112540. <https://doi.org/10.1016/j.comnet.2026.112540>.
- [4] Rice, John, and Nigel Martin. "Managing cybersecurity risks in Small businesses: A simulation-based decision framework." *Technological Forecasting and Social Change* 223 (2026): 124456. <https://doi.org/10.1016/j.techfore.2025.124456>.
- [5] Canedo, Edna Dias, Stefano Luppi Spósito, Laerte Peotta, Rafael Rabelo Nunes, and Marcelo Ladeira. "Cybersecurity risks, DevSecOps challenges, and emerging security priorities: An empirical study across Brazilian organizations." *Computers & Security* (2026): 105026. <https://doi.org/10.1016/j.cose.2026.105026>.
- [6] Cinque, Marcello, Domenico Cotroneo, and Antonio Pecchia. "Challenges and directions in security information and event management (SIEM)." In *2018 IEEE international symposium on software reliability engineering workshops (ISSREW)*, pp. 95-99. IEEE, 2018. <https://doi.org/10.1109/SIEM.2018.8539170>.
- [7] Adouani, N., T. Franco, S. Kadhi, J. Leonard, D. Co, and N. Kuhnert. "Security Incident Response for the Masses - Scalable, Open Source and Free Solutions." TheHive Project (2020). <https://thehive-project.org/>.
- [8] Olukoya, Oluwafemi. "Distilling blockchain requirements for digital investigation platforms." *Journal of Information Security and Applications* 62 (2021): 102969. <https://doi.org/10.1016/j.jisa.2021.102969>.

- [9] Tsirakis, Orestis, Konstantinos Fysarakis, Vasileios Mavroeidis, and Ioannis Papaefstathiou. "Operationalizing cybersecurity knowledge: design, implementation & evaluation of a knowledge management system for cacao playbooks." *Computers & Security* (2025): 104696. <https://doi.org/10.1016/j.cose.2025.104696>.
- [10] VS, Devi Priya, Sibi Chakkaravarthy Sethuraman, and Muhammad Khurram Khan. "Container security: precaution levels, mitigation strategies, and research perspectives." *Computers & Security* 135 (2023): 103490. <https://doi.org/10.1016/j.cose.2023.103490>.
- [11] Balasubramanian, Prasasthy, Sadaf Nazari, Danial Khosh Kholgh, Alireza Mahmoodi, Justin Seby, and Panos Kostakos. "A cognitive platform for collecting cyber threat intelligence and real-time detection using cloud computing." *Decision Analytics Journal* 14 (2025): 100545. <https://doi.org/10.1016/j.dajour.2025.100545>.
- [12] Hashmat, Fabiha, Syed Ghazanfar Abbas, Sadaf Hina, Ghalib A. Shah, Taimur Bakhshi, and Waseem Abbas. "An automated context-aware IoT vulnerability assessment rule-set generator." *Computer Communications* 186 (2022): 133-152. <https://doi.org/10.1016/j.comcom.2022.05.012>.
- [13] Yalda, Rouwa, Arshad Khan, and Narayan Nepal. "B-PhishQR-A blockchain-based framework for secure QR code verification against phishing attacks." *Telematics and Informatics Reports* (2026): 100289. <https://doi.org/10.1016/j.teler.2026.100289>.
- [14] Kampourakis, Vyrion, Vasileios Gkioulos, and Sokratis Katsikas. "A step-by-step definition of a reference architecture for cyber ranges." *Journal of Information Security and Applications* 88 (2025): 103917. <https://doi.org/10.1016/j.jisa.2024.103917>.
- [15] Kumari, Shalini, and Geeta Yadav. "Vuln2Action: An LLM-based framework for generating vulnerability reproduction steps and mapping exploits." *Journal of Information Security and Applications* 99 (2026): 104420. <https://doi.org/10.1016/j.jisa.2026.104420>.
- [16] d'Ambrosio, Nicola, Claudio Lista, Gaetano Perrone, and Simon Pietro Romano. "SMASH: An SDN-MTD framework for efficient honeypot deployment and insider threat mitigation." *Computer Networks* 269 (2025): 111327. <https://doi.org/10.1016/j.comnet.2025.111327>.
- [17] Aranguren, Mikel Egaña, Jesualdo Tomás Fernández-Breis, Bidane Leon Balentzia, Markus Rompe, and Alexander García Castro. "A comprehensive view of software vulnerability risks through enterprise knowledge graphs." *Computers & Security* (2026): 104815. <https://doi.org/10.1016/j.cose.2026.104815>.
- [18] Glas, Magdalena, Gerhard Messmann, and Günther Pernul. "Complex yet attainable? An interdisciplinary approach to designing better cyber range exercises." *Computers & Security* 144 (2024): 103965. <https://doi.org/10.1016/j.cose.2024.103965>.
- [19] Roha, Seonghyun, and Tae-Sung Kimb. "Knowledge-Grounded Threat Scenario Generation for Critical Infrastructure Security Using Open-Source Large Language Models: An Empirical Evaluation Across Communication, Energy, and Healthcare." *Expert Systems with Applications* (2026): 133328. <https://doi.org/10.1016/j.eswa.2026.133328>.
- [20] Casoria, Pellegrino, and Simon Pietro Romano. "BabeLLM: A Unified Taxonomy for NLP-based LLM Cybersecurity Applications." *Expert Systems with Applications* (2026): 133540. <https://doi.org/10.1016/j.eswa.2026.123456>.
- [21] Morchid, Abdennabi, Rachid Jebabra, Abdulla Ismail, Haris M. Khalid, Rachid El Alami, Hassan Qjidaa, and Mohammed Ouazzani Jamil. "IoT-enabled fire detection for sustainable agriculture: A real-time system using flask and embedded technologies." *Results in Engineering* 23 (2024): 102705. <https://doi.org/10.1016/j.rineng.2024.103683>.
- [22] Alsirhani, Amjad, Sijjad Ali, Mamoona Humayun, Shafique U. Rehman, and Mona Saleh Alzahrani. "NLP in Cybersecurity: Advances, Applications, and Future Directions." *Neurocomputing* 699 (2026): 134365. <https://doi.org/10.1016/j.neucom.2026.134365>.
- [23] Alhasnawi, Bilal Naji, Abdellatif M. Sadeq, Raad Z. Homod, Firas Faeq K. Hussain, Vladimír Soběslav, and Vladimír Bureš. "An extensive examination of cyberattacks, cybersecurity, and energy management in smart grid, including new advancements and machine learning." *Energy Conversion and Management: X* (2025): 101471. <https://doi.org/10.1016/j.ecmx.2025.101471>.
- [24] Campoverde-Molina, Milton, and Sergio Luján-Mora. "Cybersecurity in smart agriculture: A systematic literature review." *Computers & Security* 150 (2025): 104284. <https://doi.org/10.1016/j.cose.2024.104284>.